AMES GRANT
IN-64-CR
309747
P-12

The simultaneous integration of many trajectories using nilpotent normal forms

Matthew A. Grayson* Robert Grossman†

May, 1990

1 Introduction

Taylor's formula shows how to approximate a certain class of functions by polynomials. The approximations have two nice properties. They are arbitrarily good in some neighborhood whenever the function is analytic and they are easy to compute. Our goal is to give an efficient algorithm to approximate a neighborhood of the configuration space of a dynamical system by a nilpotent, explicitly integrable dynamical system. For a class of dynamical systems analogous to the analytic functions, this approximation will be arbitrarily good in some fixed neighborhood and easy to compute.

In [2], we give an algorithm which given a rank r yields two vector fields E_1 and E_2 on $\mathbf{R}^N$ with the properties

1. The vector fields E_1 and E_2 generate a Lie algebra isomorphic to the free, nilpotent Lie algebra on 2 generators of rank r. Let n denote the dimension of this Lie algebra.

2. If E_i, for $i = 1, \ldots, n$ denotes the vector fields corresponding to the Hall basis of a free nilpotent Lie algebra, *i.e.* $E_3 = [E_2, E_1]$, $E_4 = [E_3, E_1]$, $E_5 = [E_3, E_2]$, *etc.*, then

$$E_i(0) = \frac{\partial}{\partial x_i}, \quad i = 1, \ldots, n.$$

*IBM T.J. Watson Research Center, P.O. Box 218, Yorktown Heights NY 10598

†Laboratory for Advanced Computing, Department of Mathematics, Statistics, and Computer Science, Mail Code 249, University of Illinois, Chicago IL 60680, grossman@uicbert.eecs.uic.edu. This research was supported in part by the grants NASA NAG2-513 and NSF DMS-8904740.

(NASA-CR-187318) THE SIMULTANEOUS INTEGRATION OF MANY TRAJECTORIES USING NILPOTENT NORMAL FORMS (IBM) 12 p CSCL 12A

N91-13950

Unclas
G3/64 0309747

3. The trajectory $t \to y(t)$ satisfying

$$\dot{y}(t) = u_1(t)E_1(y(t)) + u_2(t)E_2(y(t)), \quad y(0) = y^0 \in \mathbf{R}^n$$

can be written explicitly in terms of quadratures involving the functions $t \to u(t)$.

Let $\mathcal{E}$ denote the configuration space for this system. We sometimes refer to this system as the model system.

Suppose we are given an arbitrary system of the form

$$\dot{x}(t) = u_1(t)F_1(x(t)) + u_2(t)F_2(x(t)),$$

where $x(t) \in \mathbf{R}^k$, F_1 and F_2 are vector fields on $\mathbf{R}^k$, and $t \to u_i(t)$ are given measurable controls. Let $\mathcal{F}$ denote the configuration space for this system. In order to approximate this control system to m^{th}-order, we compute the iterated Lie brackets F_i corresponding to the Hall basis E_i, obtained by substituting F_1 and F_2 for E_1 and E_2.

Our main construction is an approximating map λ which maps the configuration space $\mathcal{E}$ of the nilpotent model to the space $\mathcal{F}$ with the property that the images under λ of trajectories in $\mathcal{E}$ with measurable, bounded controls stay close to their counterparts in $\mathcal{F}$, provided that the F_i satisfy a kind of analyticity. The exact analyticity we require is described at the beginning of § 3.

The map λ turns out to be a polynomial map from $\mathbf{R}^n$ to $\mathbf{R}^k$. It depends only upon the vector fields E_i and F_i and not upon the particular controls $u_j(t)$. For this reason, it can be precomputed and used to compute efficiently a tubular neighborhood of trajectories around a given reference trajectory $x(t)$.

Nilpotent Lie algebras have been an important tool in control systems beginning with the work of Krener and Hermes, see [7], [4], [5], and [1]. The point of view of these papers was to use nilpotent Lie algebras in order to obtain theoretical results about properties of control systems. The point of view here is to focus on the some of the computational aspects of using nilpotent normal forms. In particular, we give an efficient algorithm to compute the map λ mapping trajectories of a model nilpotent system to a given system and an algorithm to integrate a tubular neighborhood of trajectories around a given fixed reference trajectory.

Section 2 defines the approximating map λ. Section 3 defines the analyticity required for our algorithm. Sections 4 and 5 state and prove the

main theorem. Section 6 shows how to apply the main theorem to integrate simultaneously many trajectories around a given fixed reference trajectory. The final section contains some examples. The appendix contains the Mathematica code we used to compute the examples.

For unexplained terminology involving Lie algebras and Hall bases, see [3] and [6].

2 An approximating map

We begin with an informal description of the main idea. The vector fields E_i define a map $\Phi_E: \mathbf{R}^n \to \mathbf{R}^n$ by $(s_1, \ldots, s_n) \to \exp(\sum_{i=1}^n s_i E_i)$. This takes the tangent space $T(\mathbf{R}^n)$ and flows it out into $\mathbf{R}^n$. $\Phi_F: \mathbf{R}^n \to \mathbf{R}^k$ is defined similarly, by exponentiating $\sum s_i F_i$. The lambda map is then $\Phi_F \circ \Phi_E^{-1}$. The map Φ_E is always invertible, so λ is always defined. But since Φ_F may be non-invertible, for instance, if $k < n$, then λ may also be non-invertible.

Computing Φ_F is usually just as hard as solving for an arbitrary trajectory, so we take approximations of Φ_F instead. Since our approximation is supposed to be good to order m, we use m applications of a Picard iteration scheme. To 0^{th} order, x^0 is 0. Substituting this into $\dot{x}^1(r) = \sum s_i F_i(0)$ we get the first order approximation $x^1(r) = r \sum s_i F_i$. Next, we would like to solve $\dot{x}^2(r) = \sum s_i F_i(x^1(r))$. This is not usually solvable explicitly, but as we're only interested in a second order approximation to the solution, we can take a first order power series expansion in the flow parameter r. Thus, we solve $\dot{x}^2(r) = a_0(t) + a_1(t)r$. We repeat this process and, eventually, get $x^m(1)$ to agree with Φ_F to order m.

More precisely, to state and prove our main theorem requires the following four definitions.

Definition 1 *For fixed $s_1, \ldots, s_n$, write $x(r; s_1, \ldots, s_n)$ for the solution $x(r)$ of $\dot{x}(r) = \sum s_i F_i(x(r))$, $x(0) = 0$, and define operators Ψ, Ψ^j by*

$$\Psi(x(r; s_1, \ldots, s_n)) = \int_0^r \sum_i s_i F_i(x(r; s_1, \ldots, s_n))\, dr$$

$$\Psi^j(x(r; s_1, \ldots, s_n)) = \int_0^r T^j\Big(\sum_i s_i F_i(x(r; s_1, \ldots, s_n))\Big)\, dr,$$

where T^j represents the j^{th}-order Taylor approximation with respect to the variable r.

Definition 2 *The trajectories $x^j(r; s_1, \ldots, s_n)$ are defined inductively:*

$$x^0(r; s_1, \ldots, s_n) = 0, \quad x^{j+1}(r; s_1, \ldots, s_n) = \Psi^j(x^j(r; s_1, \ldots, s_n)).$$

Definition 3 *The map $\Phi_F^m(s_1, \ldots, s_n): \mathbf{R}^n \to \mathbf{R}^n$ is defined as the time one flow of the trajectory x^m, namely $x^m(1; s_1, \ldots, s_n)$.*

Definition 4 *The m^{th}-order approximation to the λ map, λ^m, is defined by*

$$\lambda^m = \Phi_F^m \circ \Phi_E^{-1}.$$

3 The generalized Baker-Campbell-Hausdorff formula

Proving convergence of the algorithm, requires generalizing the Baker-Campbell-Hausdorff formula (BCH). The BCH formula writes the product of two exponentials (the composition of two flows) as the exponential of a series in the brackets (a constant flow involving the higher order brackets of the vector fields). A trajectory in a control system is a limit of compositions of piecewise constant flows, and we can use the BCH to derive a constant flow involving a series in the higher brackets which arrives at the same point. At the formal level and for systems whose vector fields generate nilpotent Lie algebras, BCH holds exactly, providing computable "geodesic normal coordinates". To approximate trajectories in other control systems, we must assume that BCH converges for them as well. We give an example later where BCH does not converge, but the vector fields are not analytic. To our knowledge, no general criteria are known which imply convergence of BCH.

We now introduce the analyticity requirement we need in order to prove convergence of our algorithm. We say that two vector fields are *BCH analytic* in case there is a δ, with $0 < \delta < 1$, such that any Lie algebra elements X_1 and X_2 in the Lie algebra generated by the F_i of the form $|X_1| < |X_2| < \delta$, satisfy the following estimate

$$|e^{X_2} e^{X_1} - e^{c_1(X_1,X_2)+\ldots+c_m(X_1,X_2)}| < \alpha^m |X_1||X_2|^m, \qquad (1)$$

where $c_i(X_1, X_2)$ are all the terms of weight i in the BCH formal expansion and α is a positive constant depending on the Lie algebra. The term $\alpha^m |X_1||X_2|^m$ comes from combining the fact that a converging series is geometric in a smaller ball, and hence a bound on the next term is comparable to the error, and the fact that every non-vanishing Lie element of weight $> m$ has at least one X_1 (the smaller) factor in every term.

Varadarajan [8] shows that the estimate $|[X,Y]| < M|X||Y|$, for all X, Y in the Lie algebra generated by F_1 and F_2, implies Condition (1). Note that this estimate holds if F_1 and F_2 generate a finite dimensional Lie algebra.

Lemma 5 *Given a positive integer m, let n be the dimension of the free nilpotent Lie algebra of rank m on two generators. Given $u_i(t)$ and $[0,T]$ as above, then there exist constants $s_1,\ldots,s_n$ such that given vector fields F_1 and F_2 as above, and the higher brackets $F_3,\ldots,F_n$ corresponding to the Hall basis elements, then the trajectories*

$$\dot{x}(t) = u_1(t)F_1(x) + u_2(t)F_2(x), \quad x(0) = 0$$

and

$$\dot{z}(t) = s_1F_1(z) + \ldots + s_nF_n(z), \quad y(0) = 0$$

satisfy $|x(T) - y(T)| < (\alpha T)^{m+1}$.

Note that the s_i depend only on the u_i and not on the F_i. The case $u_1(t) = u_2(t) = T = 1$ is the standard BCH formula.

Proof. Let $v_1(t)$ and $v_2(t)$ be step-function approximations to $u_1(t)$ and $u_2(t)$, respectively, chosen well enough so that the trajectory with the $v_i(t)$ as controls stays t^{m+1} close to $x(t)$ over the interval $[0,T]$. Suppose that the $v_i(t)$ are constant except at the times $(t_1,\ldots,t_N)$. To start, consider the flow $z^1(t)$ with controls $v_i(t)$ on the time interval $[0,t_1]$. Trivially, there is a flow $w^1(t)$ with constant coefficients $\sum_{i=1}^n s_i^1 F_i$ such that $w^1(t_1) = z^1(t_1)$, namely $s_i^1 = v_i$. The flow $z^2(t)$ is the flow $w^1(t)$ for $t \in [0,t_1]$ followed by the flow with controls $v_i(t)$ for $t \in (t_1,t_2]$. By the BCH convergence condition (1), there is a flow $w^2(t)$ with different constant coefficients $\sum_{i=1}^n s_i^2 F_i$ such that $|w^2(t_1) - z^2(t_1)| < (t_2 - t_1)(\alpha T)^m$. At stage j, the error is $< (t_{j+1} - t_j)(\alpha T)^m$, and so the total error is $< (\alpha T)^m \sum(t_{j+1} - t_j) = (\alpha T)^{m+1}$ as desired. ∎

Since the BCH formula holds exactly in the configuration space E, the trajectory $z(t)$ is exactly $\lambda(y(t))$ in this case.

4 The Picard-Taylor Method

Recall that the map Φ_F^m is the easily computed Picard-Taylor approximation of the exponential map $\Phi_F(s) = \sum s_i F_i$. In this section, we prove that the error introduced by using Φ_F^m is $O(s^{m+1})$, just as in Taylor's theorem.

The convergence of a Picard iteration scheme depends on a bound B and a Lipschitz constant L for the vector fields: $|F_i(p)| < B$ and $|F_i(p) - F_i(q)| <$

$L|p-q|$ for p and q in some ball about the origin. The choice of ball determines B, L and a fixed time T', such that the approximate trajectories do not leave the ball where these estimates hold.

Lemma 6 *Given F_i as above, there is a constant $\delta_2 > 0$ and a positive constant C such that $|(\Phi_F^m - \Phi_F)(s)| < C|s|^{m+1}$ whenever $|s| < \delta_2$.*

Proof. Let $z^0(r; s_1, \ldots, s_n)) = 0$ and $z^{j+1}(r; s_1, \ldots, s_n)) = \Psi(z^j)$. Examine the convergence of $z^j(1; s_1, \ldots, s_n)$ to $\Phi_F(s) = \exp(\sum_i s_i F_i)$ as $j \to \infty$. Assume, as an induction hypothesis, that $z_1^j(r; s)$ and $z_2^j(r; s)$ are two trajectories which are close in the sense that $|z_1^j(r; s) - z_2^j(r; s)| < c(r|s|)^j$ for $r \in [0, 1]$. Then

$$\begin{aligned} |\Psi(z_1^j(r;s)) - \Psi(z_2^j(r;s))| &\leq \int_0^r |\sum_i s_i (F_i(z_1^j(r;s)) - F_i(z_2^j(r;s)))|\, dr \\ &\leq c(r|s|)^{j+1} L/(j+1), \end{aligned}$$

and so there is some fixed constant C such that

$$|z^j(1;s) - \Phi_F(s)| \leq C|s|^{j+1},$$

as desired. In particular, we have

$$|z^m(1,s) - \Phi_F(s)| \leq C|s|^{j+1}.$$

It remains to estimate $|x^{j+1}(r;s) - z^{j+1}(r;s)|$. Let

$$|x^0(r;s) - z^0(r;s)| = 0, \quad x^{j+1}(r;s) = \Psi^j(x^j(r;s)).$$

Assuming that $|x^j(r;s) - z^j(r;s)| < C(r|s|)^{j+1}$, we estimate

$$\begin{aligned} |x^{j+1}(r;s) - z^{j+1}(r;s)| &= |\Psi^j(x^j(r;s)) - \Psi(z^j(r;s))| \\ &\leq |\Psi^j(x^j(r;s)) - \Psi^j(z^j(r;s))| \\ &\quad + |\Psi^j(z^j(r;s)) - \Psi(z^j(r;s))| \\ &\leq \int_0^r |T^j(\sum_i s_i F_i(x^j(r;s)) - T^j \sum_i s_i F_i(z^j(r;s)))|\, dr \\ &\quad + \int_0^r |T^j(\sum_i s_i F_i(z^j(r;s)) - \sum_i s_i F_i(z^j(r;s)))|\, dr \\ &\leq \int_0^r |T^j(\sum_i s_i F_i(x^j(r;s)) - \sum_i s_i F_i(z^j(r;s)))|\, dr \\ &\quad + \int_0^r |C \sum_i s_i r^{j+1}|\, dr \\ &\leq C\frac{(r|s|)^{j+2}}{j+2}. \end{aligned}$$

In particular, we have

$$|x^m(1,s) - z^m(1,s)| \leq C|s|^{j+1}.$$

Since $\Phi_F^m(s) = x^m(1;s)$, the lemma follows. ∎

5 The main theorem

In this section, we combine the lemmas of the two previous sections to prove that the easily computed approximating map λ^m maps trajectories of the explicitly integrable system $\mathcal{E}$ to trajectories of the arbitrary system $\mathcal{F}$ with error $O(s^{m+1})$.

Theorem 7 . *Given a positive constant M and δ, a positive integer m, and a positive time T, there exists a positive constant C such that, given measurable controls $u_1(t)$ and $u_2(t)$ satisfying $|u_i(t)| < M$ for all $t \in [0,T]$, vector fields F_1 and F_2 satisfying the BCH analyticity condition (1), and solutions $x(t)$ and $y(t)$ of*

$$\begin{aligned} \dot{x}(t) &= u_1(t)F_1(x(t)) + u_2(t)F_2(x(t)), \quad x(0) = 0 \\ \dot{y}(t) &= u_1(t)E_1(y(t)) + u_2(t)E_2(y(t)), \quad y(0) = 0, t \in [0,T], \end{aligned}$$

then $|\lambda^m(y(t)) - x(t)| < (Ct)^{m+1}$ for $t \in [0,T]$.

Proof. It is sufficient to show that the estimate holds at time T. Applying Lemma 5 twice, once to the system $\mathcal{E}$ and once to the system $\mathcal{F}$ shows that

$$|x(T) - \lambda(y(T))| \leq (\alpha T)^{m+1}.$$

Also, we know from Lemma 6 that

$$|x^m(1;s) - x(1;s)| \leq C|s|^{m+1}.$$

Since $\lambda^m(y(T)) = x^m(1;s)$ and $|s| <$ constant $\cdot\, T$, the theorem follows. ∎

6 Simultaneous Integration of Trajectories

In this section, we describe an algorithm to integrate simultaneously a neighborhood of trajectories around a given fixed trajectory. Fix controls u_1 and u_2 and an initial condition $x^0 \in \mathcal{F}$, and let $x(t)$ denote the corresponding trajectory of the system $\mathcal{E}$. We give an algorithm yielding p trajectories in a tubular neighborhood around the fixed reference trajectory $x(t)$.

Step 1. Solve the equation $\lambda(y) = x^0$ for y. Let $y^0 \in \mathcal{F}$ denote the root.

Step 2. For points $y^1, \ldots, y^p$ in a neighborhood of y^0, compute the trajectory $y^j(t)$ satisfying

$$\dot{y}(t) = u_1(t)E_1 + u_2(t)E_2, \quad y(0) = y^j.$$

Step 3. For each point $y^j(t)$ along the trajectory, compute the corresponding point $\lambda^m(y^j(t))$, for $j = 1, \ldots, p$.

The following theorem follows immediately from Theorem 7.

Theorem 8 *For points y^j sufficiently close to y^0 and small enough time t, the trajectories $\lambda^m(y^j(t))$ all lie within a fixed tubular neighborhood of the reference trajectory $x(t)$. These trajectories can be computed by simply evaluating the map λ^m, which can be precomputed, at different points.*

7 Examples

We begin by showing why we require a condition such as BCH analyticity for the vector fields F_i.

Let

$$X = \frac{\partial}{\partial x_1}, \quad Y = \frac{\partial}{\partial x_2} - \phi(x_1, x_2)\frac{\partial}{\partial x_3},$$

where $\phi(x_1, x_2)$ is identically 1 if $|x_1 - x_2| > 1/4$, and it is smooth and between 0 and 1 elsewhere, positive away from the diagonal, and vanishing to infinite order on the diagonal. Now the x_3 coordinate of $\exp(aY)\circ\exp(aX)$ is negative for all $a > 0$. But all brackets of X and Y vanish on the diagonal. Therefore, no formula of the form $\exp(aX)\circ\exp(aY) = exp(X+Y+$ brackets in X and Y can be valid.

Figures 1 and 2 contain the Mathematica code we used to test the algorithm. Figure 3 contains the vector fields E_i and F_i and the λ^m map. Note that the map λ^m is a polynomial map. Figure 4 contains the result of flowing along the explicitly integrable flow in the $\mathcal{E}$ space, applying the map λ^m, and comparing the result to flowing in the $\mathcal{F}$ space using a Runge Kutta flow.

```
Brac[v_,w_]:=Block[{i,j},Table[Sum[v[[i]] D[w[[j]],x[i]]
   -w[[i]] D[v[[j]],x[i]],{i,Length[v]}] ,{j,Length[v]}] ]

Flow[p_,v_,{rt_,r0_,r1_}]:= Block[{xt,v1,x1,i,j,n},
   n=Length[p];
   v1=v;
   xt=p;
   v1[[1]]=v[[1]];
   For[i=1,i<n,i++,
      xt[[i]] = Integrate[v1[[i]],rt]+p[[i]];
      v1[[i+1]] = v[[i+1]]/.Table[x[j]->xt[[j]],{j,1,i}]
      ];
   xt[[n]] = Integrate[v1[[n]],rt]+p[[n]];
   (xt/.{rt->r1})-(xt/.{rt->r0})
]
Flow[p_,v_,{rt_,r1_}]:=Flow[p,v,{rt,0,r1}]
Flow[p_,v_,{rt_}]:=Flow[p,v,{rt,0,1}]
Flow[p_,v_]:=Flow[p,v,{rt,0,1}]
```

Figure 1: Mathematica code to compute brackets and integrable flows.

```
Picard[f_,m_]:=Block[{a,i,j,r,n},
   n=Length[f];                  (* n = dimension *)
   a=f/.Table[x[i]->0,{i,n}];    (* substitue r=0 *)
   For[j=1,j<=m,j++,             (* Loop through m times... *)
      a=Integrate[a,r];
      a=f/.Table[x[i]->a[[i]],{i,n}]; (* substitue *)
      (* approximate by something integrable   *)
      (* at increasing accuracy *)
      a=Table[Normal[Series[a[[i]],{r,0,j-1}]],{i,n}]
   ];
   Integrate[a,{r,0,1}] (* Get the final answer *)
]

Lambda[e_,f_,n_,m_]:=Block[{i,phiF,t,phiE,phiEInv},
   (* e, f are functions whose values are vector fields *)
   (* n is the number of vf's, m is the degree of approx. *)
   (* Get the F flow by Picard iteration *)
   phiF=Picard[Sum[t[i]f[i],{i,n}],m];
   (* Get the E flow by symbolic integration *)
   phiE=Flow[Table[0,{i,n}],Sum[t[i]e[i],{i,n}]];
   (* Invert the E flow symbolically *)
   phiEInv=Solve[Table[b[i]==phiE[[i]],{i,n}],
      Table[t[i],{i,n}]][[1]];
   (* Form the composition *)
   phiF/.phiEInv
]
```

Figure 2: Mathematica code to compute the map λ^m.

```
e[1]={1,0,0}
e[2]={0,1,-x[1]}
e[3]=Brac[e[2],e[1]]
f[1]={Sin[x[3]],0,x[2]Cos[x[3]]}
f[2]={0,Cos[x[3]],x[1]Sin[x[3]]}
f[3]=Brac[f[2],f[1]]

L3=Lambda[e,f,3,3];
```

```
     2
 b[1]  b[2]   b[1] (b[1] b[2] + 2 b[3])
{---------- + ------------------------,
     6                    4

                                      2
         b[2] (b[1] b[2] + 2 b[3])
  b[2] + -------------------------,
                    24

                                                    3
  b[1] b[2]   b[1] b[2] + 2 b[3]   (b[1] b[2] + 2 b[3])
  --------- + ----------------- - --------------------}
      2              2                    12
```

Figure 3: The vector fields and the λ^m map.

```
Flow[{0,0,0},(1+t+t^2+t^3)e[1]+(Sin[20t])e[2],{t,.1}]

{0.105358, 0.0708073, -0.0045167}

L3/.Table[b[i]->%[[i]],{i,3}]

{0.0000895593, 0.0708073, 0.00294345}

RungeKuttaHiD[Join[{1},(1+t+t^2+t^3)f[1]+(Sin[20t]) f[2]],
{t,x[1],x[2],x[3]},{0,0,0,0},0.02,5]

{0.1, 0.0000842708, 0.0708076, 0.00294349}
```

Figure 4: Comparing the map λ^m and a Runge Kutta flow.

References

[1] P. E. Crouch, "Dynamical realizations of finite Volterra series," *SIAM J. Control Optim.*, 19 (1981), pp. 177–202.

[2] M. Grayson and R. Grossman, "Models for free, nilpotent lie algebras," *J. Algebra*, to appear.

[3] M. Hall, "A basis for free Lie rings and higher commutators in free groups," *Proc. Amer. Math. Soc.*, 1 (1950), pp. 575–581.

[4] H. Hermes,"Control systems which generate decomposible Lie algebras," *J. Diff. Eqns.*, 44 (1982), pp. 166–187.

[5] H. Hermes, "Nilpotent approximations of control systems and distributions," *SIAM J. Control Optim.*, 24 (1986), pp. 731–736.

[6] N. Jacobson, *Lie Algebras*, John Wiley and Sons, New York, 1962.

[7] A. Krener, "Bilinear and nonlinear realizations of input-output maps," *SIAM J. Control Optim.*, 13 (1975), pp. 827–834.

[8] V. S. Varadarajan, *Lie Groups, Lie Algebras, and their Representations*, Prentice-Hall, Inc., Englewood Cliffs, 1974.